```
# -*- coding:utf-8 -*-
import tkinter
import tkinter.messagebox
import random

# キャンバスの横方向・縦方向のサイズ（px）
CANVAS_SIZE = 400

# ラインの数
NUM_LINE = 10

# 色の設定
BOARD_COLOR = "burlywood3" # 盤面の背景色
YOUR_COLOR = 'black' # あなたの石の色

# プレイヤーを示す値
YOU = 1

class Gobang():
    def __init__(self, master):
        '''コンストラクタ'''
        self.master = master # 親ウィジェット
        self.player = YOU # 次に置く石の色
        self.board = None # 盤面上の石を管理する 2 次元リスト
        self.color =  YOUR_COLOR
        self.nextDisk = None

        # ウィジェットの作成
        self.createWidgets()

        # イベントの設定
        self.setEvents()

        # 五目並べゲームの初期化
        self.initGobang()


    def createWidgets(self):
        '''ウィジェットを作成・配置する'''
        # キャンバスの作成
        self.canvas = tkinter.Canvas(
            self.master,
```

```
43          bg=BOARD_COLOR,
44          width=CANVAS_SIZE,
45          height=CANVAS_SIZE,
46          highlightthickness=0
47      )
48      self.canvas.pack(padx=10, pady=10)
49
50  def setEvents(self):
51      '''イベントを設定する'''
52      # キャンバス上のマウスクリックイベントを受け付ける
53      self.canvas.bind('<ButtonPress>', self.click)
54
55
56  def initGobang(self):
57      '''ゲームの初期化を行う'''
58      # 盤面上の石を管理する２次元リストを作成（最初は全てNone）
59      self.board = [[None] * (NUM_LINE) for i in range(NUM_LINE)]
60
61      # 線と線の間隔（px）を計算
62      self.interval = CANVAS_SIZE // (NUM_LINE + 1)
63
64      # 交点描画位置の左上オフセット
65      self.offset_x = self.interval
66      self.offset_y = self.interval
67
68      # 縦線を描画
69      for x in range(NUM_LINE):
70
71          # 線の開始・終了座標を計算
72          xs = x * self.interval + self.offset_x
73          ys = self.offset_y
74          xe = xs
75          ye = (NUM_LINE - 1) * self.interval + self.offset_y
76
77          # 線を描画
78          self.canvas.create_line(
79              xs, ys,
80              xe, ye,
81          )
82
83      # 横線を描画
84      for y in range(NUM_LINE):
```

```
85
86          # 線の開始・終了座標を計算
87          xs = self.offset_x
88          ys = y * self.interval + self.offset_y
89          xe = (NUM_LINE - 1) * self.interval + self.offset_x
90          ye = ys
91
92          # 線を描画
93          self.canvas.create_line(
94              xs, ys,
95              xe, ye,
96          )
97
98      def drawDisk(self, x, y, color):
99          '''(x,y)の交点に色がcolorの石を置く（円を描画する）'''
100         # (x,y)の交点の中心座標を計算
101         center_x = x * self.interval + self.offset_x
102         center_y = y * self.interval + self.offset_y
103
104         # 中心座標から円の開始座標と終了座標を計算
105         xs = center_x - (self.interval * 0.8) // 2
106         ys = center_y - (self.interval * 0.8) // 2
107         xe = center_x + (self.interval * 0.8) // 2
108         ye = center_y + (self.interval * 0.8) // 2
109
110         # 円を描画する
111         tag_name = 'disk_' + str(x) + '_' + str(y)
112         self.canvas.create_oval(
113             xs, ys,
114             xe, ye,
115             fill=YOUR_COLOR,
116
117         )
118
119         return tag_name
120
121     def getIntersection(self, x, y):
122         '''キャンバス上の座標を交点の位置に変換'''
123         ix = (x - self.offset_x + self.interval // 2) // self.interval
124         iy = (y - self.offset_y + self.interval // 2) // self.interval
125
126         return ix, iy
```

```
127
128     def click(self, event):
129         '''盤面がクリックされた時の処理'''
130         # クリックされた位置がどの交点であるかを計算
131         x, y = self.getIntersection(event.x, event.y)
132         if x < 0 or x >= NUM_LINE or y < 0 or y >= NUM_LINE:
133             # 盤面外の交点の場合は何もしない
134             return
135 
136         if not self.board[y][x]:
137             # 石が置かれていない場合はクリックされた位置に石を置く
138 
139             # 石を置く
140             self.place(x, y, self.color[self.player])
141 
142     def place(self, x, y, color):
143         '''(x,y)の交点に色がcolorの石を置く'''
144         # (x,y)に石を置く（円を描画する）
145         self.drawDisk(x, y, color)
146 
147         # 描画した円の色を管理リストに記憶させておく
148         self.board[y][x] = color
149 
150 
151 # スクリプト処理ここから
152 app = tkinter.Tk()
153 app.title('五目並べ')
154 gobang = Gobang(app)
155 app.mainloop()
156 
157 
158 
```